# SOFTWARE COMMUNICATIONS ARCHITECTURE SPECIFICATION

# APPENDIX E-1: PLATFORM SPECIFIC MODEL (PSM) - COMMON OBJECT REQUEST BROKER ARCHITECTURE (CORBA)



01 October 2012
Version: 4.0.1

Prepared by:

**Joint Tactical Networking Center (JTNC)**
33000 Nixie Way
San Diego, CA 92147-5110

Statement A - Approved for public release; distribution is unlimited (18 November 2013)

# REVISION SUMMARY

| Version | Revision | Date |
|---|---|---|
| Next <Draft> | Initial Draft Release | 30 November 2010 |
| Candidate Release | Initial Release | 27 December 2011 |
| 4.0 | ICWG Approved Release | 28 February 2012 |
| 4.0.1 | Incorporated transition to JTNC and applied SCA 4.0 Errata Sheet v1.0 | 01 October 2012 |

# TABLE OF CONTENTS

# TABLE OF TABLES

# APPENDIX E-1     PSM – CORBA

## E-1.1 SCOPE

This appendix defines the platform specific transport and technology model using the Common Object Request Broker Architecture (CORBA).

### E-1.1.1     <u>SCA CORBA Profiles</u>

This appendix includes three SCA CORBA profiles based on CORBA/*e* [1] with additional features from RT CORBA [2].  The SCA CORBA profiles are characterized as follows:

1. SCA Full CORBA (Full) Profile – is the Full CORBA profile

2. SCA Lightweight CORBA (LW) Profile – is more constrained than the SCA Full CORBA Profile and is targeted towards environments with limited computing support.

3. SCA Ultra-Lightweight CORBA (ULW) Profile – is more constrained than the SCA Lightweight CORBA Profile and is specifically intended for processing elements with even more limited computing support.

Specifically:
- the Full Profile is intended for applications that will be hosted on most General Purpose Processor (GPP) platforms,
- the LW Profile is intended for applications that will be hosted on platforms such as DSPs,
- the ULW Profile is intended for applications hosted on both DSPs and FPGAs.

These profiles include features that have been chosen to support SCA Applications while avoiding features that require excessive processing resources. While platform designers will often know the resource availability and may choose to use resource intensive features, applications are usually intended to be portable between platforms and so it is desirable to minimize their demand on resources to ease porting to more constrained environments.

Each profile characterizes the Interface Definition Language (IDL) features allowed for definition of interfaces between application components. The LW Profile narrows the IDL feature set in order to limit the processing overhead caused by a number of types in the Full Profile. The ULW Profile narrows the constructs even further to accommodate typical limitations of Digital Signal Processor (DSP) and Field Programmable Gate Array (FPGA) environments. The shared IDL foundation of the profiles facilitates portability not only between platforms, but also across processing elements and transfer mechanisms. An example of how component portability may be enhanced is if the IDL recommendations from more constrained profiles are used when defining application interfaces targeted for components deployed within less constrained processing elements.

Some of the resource intensive features that have been omitted will reduce resource demands even with an Object Request Broker (ORB) that supports them if the features are not used. However, to achieve the full goal of reducing demand on system resources, ORBs omitting support for these features will be required.

Because platforms may use additional features, the three SCA CORBA profiles are not intended to specify complete ORBs for hosting SCA systems.

## E-1.2 CONFORMANCE

See SCA Appendix E.

## E-1.3 CONVENTIONS

Within this appendix, the following abbreviations are used:

"MAN"     indicates that the identified operation or feature is mandatory for the indicated profile

"NRQ"     indicates that the identified operation or features is not required for the indicated profile

"PRT"     indicates that only a subset of the identified operation or features unit of functionality is required.  This designation will be followed by a note or cross-reference indicating which elements are required

"MAN*"    indicates that the identified operation or feature is mandatory for the indicated profile but is discouraged for use by applications to improve portability across non-CORBA implementations

"N/A"     indicates that the identified operation or feature is not applicable for direct usage by an application

## E-1.4 NORMATIVE REFERENCES

N/A

## E-1.5 INFORMATIVE REFERENCES

The following is a list of documents referenced within this appendix or used as reference or guidance material in its development.

[1]     Common Object Request Broker Architecture (CORBA) for embedded Specification, Version 1.0 formal/2008-11-06, November 2008.

[2]     Real-time CORBA Specification, Version 1.2 formal/2005-01-04, January 2005.

[3]     Minimum CORBA Specification, Version 1.0 formal/02-08-01, August 2002.

## E-1.6 CONSTRAINTS

### E-1.6.1     Operating Environment

SCA505 The OE shall provide the features designated as mandatory, as specified in E-1.7, for the SCA CORBA profile implemented.

The OE is not limited to providing the features designated as mandatory by the SCA CORBA profile.

### E-1.6.2      <u>Applications</u>

SCA506 Applications shall be limited to using the features designated as mandatory, as specified in E-1.7, for the SCA CORBA profile implemented.

Applications are permitted to use any feature of the standard CORBA language mapping being used, unless specifically disallowed in this appendix.

### E-1.6.3      <u>Logical Devices</u>

Logical Devices are not restricted by this appendix.

### E-1.6.4      <u>Platform Services</u>

Platform Services are not restricted by this appendix.

## E-1.7 SCA CORBA PROFILE DEFINITIONS

### E-1.7.1      <u>Features from CORBA/*e*</u>

SCA507 The features included in the Full, LW and ULW Profiles listed in Attachment 1 to this appendix shall behave as described in the applicable clauses of CORBA/*e* [1].

For convenience, columns showing the features included in some other profiles (noted with an 'x') have been included: Minimum CORBA [3] and CORBA/*e* compact [1].

#### E-1.7.1.1    Complex Types in Any

The use of CORBA Any data type, such as in CF::*PropertySet* interface can have very resource intensive impact. The CF requires the use of Any in *PropertySet* interface to support properties in the domain profile.

The CF requires the Any data type to support values resulting from the data descriptions in the domain profile.

- CORBA Basic Types (CORBA/*e* [1] section 6.10.1) excluding WChar
- SEQUENCES of the above types (such as String)
- *PropertySet* or SEQUENCE of *PropertySet* as a value

The CF requires support for the value to contain a *PropertySet*, so the predefined struct for *PropertySet* is also required.

Although user-defined structs can be defined in XML, these are not sent as CORBA structs but rather as nested SEQUENCES. Therefore support for CORBA struct (other than that in PropertySet) and union is not required by the CF.

The SCA deprecates the use of other types, including user defined CORBA structs and unions in the value of *PropertySet* interface properties or other uses of Any in applications and non-standard APIs. While such use is permitted in SCA compliant applications, it is discouraged.

The Full Profile discourages the use of CORBA Any.

### E-1.7.1.2  Unchecked Narrow

CORBA language mappings typically provide a mechanism for narrowing an object reference from a base interface to a more derived interface. In addition, the CORBA/*e* [1] specification (section 9.2.7) requires that an unchecked narrow operation be provided for languages that support inheritance. It is permissible to use an implementation specific unchecked narrow operation in all of the SCA CORBA profiles if one has not been defined within the relevant standardized language mapping.

## E-1.7.2          Features from RTCORBA

This appendix permits the use of a few RT CORBA [2] features that provide useful ways to system tune performance but are not supported by CORBA/*e* [1].  SCA508 The features included in the Full, LW and ULW Profiles listed in Attachment 2 to this appendix shall behave as described in the applicable clauses of RT CORBA [2].

### E-1.7.2.1  ORB_init Parameters

The Full Profile includes methods to create certain POA policies, but these methods are only supported on child POAs. The root POA has default settings for these policies that cannot be changed during the life of the root POA. The LW Profile does not support the creation of child POAs or calls to the policy creation methods.  In some systems it is useful to use a policy other than the default even when it cannot be changed dynamically.  The creation of child POAs only to allow static policies other than the default adds undesirable overhead.  Therefore it is desirable to allow creating the root POA with policies other than the default.

SCA509 The Full and LW Profiles shall support the additional standardized parameters identified in **Table 1** to the ORB_init call to allow the root POA to be created with non-default policies.  These additional parameters are not standardized in CORBA/*e* [1].

This section is not applicable to ULW Profile because it does not include the ORB_init feature.

**Table 1: ORB_init() Parameters**

| Policy | Default Value | Alternate Value | Optional Parameter to Override | Full Profile | LW Profile | ULW Profile |
|---|---|---|---|---|---|---|
| Lifespan Policy | TRANSIENT | PERSISTENT | -ORBPOAPersistent | MAN | MAN | NRQ |
| ID Uniqueness Policy | UNIQUE_ID | MULTIPLE_ID | -ORBPOAMultipleId | MAN | MAN | NRQ |
| ID Assignment Policy | SYSTEM_ID | USER_ID | -ORBPOAUserId | MAN | MAN | NRQ |

Calls to ORB_init() will cause the root POA to be created with the default setting for each of these policies unless one or more of the optional parameters appear in the parameter list. If one or more are present then the root POA is created with the corresponding alternate value.

### E-1.7.2.2  Thread Pools

The RT CORBA [2] Thread Pools features are included in the Full Profile as a mechanism for a server to provide multiple priorities.  This mechanism provides improved system performance

with less resource use than priority banding.  For example, after startup, pool priorities can be changed to optimize for runtime.

### E-1.7.2.3   Server Priority Model

RT CORBA [2] defines two models for determining the priority for servants processing method calls:
  *   CLIENT_PROPAGATED model and
  *   SERVER_DECLARED model.

Only the CLIENT_PROPAGATED model is supported in CORBA/*e* [1].  The SERVER_DECLARED model is included in the Full Profile for managing priorities.  The SERVER_DECLARED model does not require propagating the priority with each method call and so resulting in smaller messages and less processing to set priority on each call.

### E-1.7.2.4   ServerProtocolPolicy and ClientProtocolPolicy

The RT CORBA [2] ServerProtocolPolicy and ClientProtocolPolicy features are included in the Full Profile to support lightweight methods for managing priorities.

### E-1.7.2.5   activate_object_with_priority

The RT CORBA [2] activate_object_with_priority feature is included in the Full Profile to support light weight methods for managing priorities.

## E-1.8 TRANSPORT INITIALIZATION

Some CORBA transports require transport-specific initialization using vendor specific functions. Since transport initialization is not standardized, implementation specific initialization operations are considered to be compliant within the PSM if an equivalent standardized interface does not exist within the PSM referenced standards in section E-1.4.

## E-1.9 ATTACHMENTS

This appendix includes the following:
  *   Attachment 1: SCA CORBA Profiles (from CORBA/e)
  *   Attachment 2: SCA CORBA Profiles (from RT CORBA)

These attachments include the Full/LW/ULW Profile features from CORBA/*e* [1] and RT CORBA [2].